ALU 19 – BIT :-

MAIN PROGRAM :-

module alu\_19bit (

input [18:0] a, b,

input [3:0] alu\_op,

output reg [18:0] result,

output reg zero);

always @(\*) begin

case (alu\_op)

4'b0000: result = a + b;

4'b0001: result = a - b;

4'b0010: result = a \* b;

4'b0011: result = a / b;

4'b0100: result = a & b;

4'b0101: result = a | b;

4'b0110: result = a ^ b;

4'b0111: result = ~a;

4'b1000: result = a + 19'b1;

4'b1001: result = a - 19'b1;

default: result = 19'b0;

endcase

zero = (result == 19'b0);

end

endmodule

TESTBENCH :-

`timescale 1ns / 1ps

module alu\_tb();

reg [18:0] a, b;

reg [3:0] alu\_op;

wire [18:0] result;

wire zero;

alu\_19bit uut (

.a(a),

.b(b),

.alu\_op(alu\_op),

.result(result),

.zero(zero));

initial begin

a = 19'd5; b = 19'd3; alu\_op = 4'b0000;

#10;

$display("ADD: a=%d, b=%d, result=%d, zero=%b", a, b, result, zero);

a = 19'd5; b = 19'd3; alu\_op = 4'b0001;

#10;

$display("SUB: a=%d, b=%d, result=%d, zero=%b", a, b, result, zero);

a = 19'd2; b = 19'd3; alu\_op = 4'b0010;

#10;

$display("MUL: a=%d, b=%d, result=%d, zero=%b", a, b, result, zero);

a = 19'b1111; b = 19'b1100; alu\_op = 4'b0100;

#10;

$display("AND: a=%b, b=%b, result=%b", a, b, result);

#10 $finish;

end

endmodule

OUTPUT :-

